AMENDMENTS TO THE CLAIMS

Please amend the claims as follows:

1. (Currently Amended) A method comprising:

selecting one or more microarchitecture events relating to a microprocessor executing an application process to be monitored by one or more hardware monitors;

establishing parameters regarding the monitoring of the microarchitecture events by setting one or more monitor control vectors, the one or more monitor control vectors to monitor events in a detection exception stage of an execution pipeline;

storing ~~traces that are~~ profile data that is captured by the one or more hardware monitors in a first level profile buffer, the first level profile buffer being a architecturally non-visible memory, the storing of the profile data being performed in a write-back stage of the execution pipeline;

transferring the captured ~~traces~~ profile data from the first level profile buffer to a second level profile buffer, the second level profile buffer being an architecturally visible storage;

obtaining the captured ~~traces~~ profile data from the second level profile buffer;

processing the captured ~~traces~~ profile data;

identifying a region of interest in the application process for optimization based at least in part on the captured ~~traces~~ profile data; and

optimizing the region of interest in the application process.

2. (Original) The method of claim 1, wherein setting each monitor control vector comprises setting one or more fields of the monitor control vector to control the monitoring of the microarchitecture event.

3. (Original) The method of claim 2, wherein setting the one or more fields of each monitor control vector includes setting a control field to establish the type of microarchitecture event that is monitored by a hardware monitor.

4. (Original) The method of claim 2, wherein setting the one or more fields of each monitor control vector includes setting a trigger field to control when a microarchitecture event is monitored.

5. (Currently amended) The method of claim 2, wherein setting the one or more fields of each monitor control vector includes storing a pointer in a handler field, the pointer identifying a handler routine to process captured ~~traces~~ profile data corresponding to the monitor control vector.

6. (Canceled)

7. (Currently Amended) The method of claim 1, wherein obtaining the captured ~~traces~~ profile data for a microarchitecture event from the second level profile buffer occurs when a memory buffer in the second level profile buffer that is assigned for the monitored microarchitecture event is fully allocated.

8. (Currently amended) The method of claim 7, further comprising setting one or more conditions for obtaining captured ~~traces~~ profile data when the memory buffer in the second level profile buffer is not fully allocated, and setting one or

more conditions for transferring captured ~~traces~~ profile data from the first level profile buffer to the second level profile buffer.

9. (Currently Amended) The method of claim 8, further comprising receiving an interrupt or special event handler if the memory buffer that is assigned for the microarchitecture event is fully allocated or if a condition for obtaining captured ~~traces~~ profile data when the memory buffer in the profile buffer is not fully allocated is met.

10. (Original) The method of claim 1, wherein the microarchitecture event monitored is an instruction cache miss event.

11. (Previously Presented) A machine-readable medium having stored thereon data representing instructions that, when executed by a processor, cause the processor to perform operations comprising:

selecting one or more microarchitecture events relating to a microprocessor executing an application process to be monitored by one or more hardware monitors;

establishing parameters regarding the monitoring of the microarchitecture events by setting one or more monitor control vectors, the one or more monitor control vectors to monitor events in a detection exception stage of an execution pipeline;

storing ~~traces that are~~ profile data that is captured by the one or more hardware monitors regarding the occurrence of the one or more microarchitecture events in a first level profile buffer, the first level profile buffer being a architecturally non-visible memory, the storing of the profile data being performed in a write-back stage of the execution pipeline;

transferring the captured ~~traces~~ profile data from the first level profile buffer to a second level profile buffer, the second level profile buffer being an architecturally visible storage;

obtaining the captured ~~traces~~ profile data from the second level profile buffer;

processing the captured ~~traces~~ profile data;

identifying a region of interest in the application process for optimization based at least in part on the captured ~~traces~~ profile data; and

optimizing the region of interest in the application process.

12. (Original) The medium of claim 11, wherein setting each monitor control vector comprises setting one or more fields of the monitor control vector to control the monitoring of the microarchitecture event.

13. (Original) The medium of claim 12, wherein setting the one or more fields of each monitor control vector includes setting a control field to establish the type of microarchitecture event that is monitored by a hardware monitor.

14. (Original) The medium of claim 12, wherein setting the one or more fields of each monitor control vector includes setting a trigger field to control when a microarchitecture event is monitored.

15. (Currently Amended) The medium of claim 12, wherein setting the one or more fields of each monitor control vector includes storing a pointer in a handler field, the pointer identifying a handler routine to process the captured ~~traces~~ profile data

associated with the occurrence of a microarchitecture event corresponding to the monitor control vector.

16. (Canceled)

17. (Currently Amended) The medium of claim 11, wherein obtaining the captured ~~traces~~ profile data for a microarchitecture event from the second level profile buffer occurs when a memory buffer in the second level profile buffer that is assigned for the monitored microarchitecture event is fully allocated

18. (Currently Amended) The medium of claim 17, wherein the instructions include instructions that, when executed by a processor, cause the processor to perform operations comprising setting one or more conditions for obtaining captured ~~traces~~ profile data when the memory buffer in the second level profile buffer is not fully allocated, and setting one or more conditions for transferring captured ~~traces~~ profile data from the first level profile buffer to the second level profile buffer.

19. (Currently Amended) The medium of claim 18, wherein the sequences of instructions include instructions that, when executed by a processor, cause the processor to perform operations comprising receiving an interrupt or special event handler if the memory buffer that is assigned for the microarchitecture event is fully allocated or if a condition for obtaining captured ~~traces~~ profile data when the memory buffer in the profile buffer is not fully allocated is met.

20. (Original) The medium of claim 11, wherein the microarchitecture event monitored is an instruction cache miss event.

21. (Currently Amended) A hardware assisted dynamic optimizer, comprising:

an interface to a microprocessor through which the hardware assisted dynamic optimizer establishes parameters regarding the monitoring of one or more microarchitecture events occurring during the execution of an application by the microprocessor, the monitoring to occur in a detection exception stage of an execution pipeline;

one or more handler routines, each handler routine including instructions to process profiles of a monitored microarchitecture event that are captured by the microprocessor;

a first level profile buffer to initially store captured profiles, the first level profile buffer being architecturally non-visible, the storing of the captured profiles being performed in a write-back stage of the execution pipeline;

a second level profile buffer to receive captured profiles from the first level profile buffer, the second level profile buffer being architecturally visible; and

one or more optimizers, each optimizer including instructions for optimizing a section of the application, the section of the application being chosen by the hardware assisted dynamic optimizer at least in part based on the captured profiles of a monitored microarchitecture event.

22. (Original) The hardware assisted dynamic optimizer of claim 21, wherein each monitor control vector includes a plurality of fields to control the monitoring of

the microarchitecture event, the plurality of fields being set by the hardware assisted dynamic optimizer.

23. (Original) The hardware assisted dynamic optimizer of claim 22, wherein the plurality of fields includes:

a control field to establish the type of microarchitecture event that is monitored,

a trigger field to control when the microarchitecture event is monitored, and

a handler field to store a pointer to the handler routine for the microarchitecture event.

24. (Original) The hardware assisted dynamic optimizer of claim 21, wherein optimizing a section of the application includes increasing the speed of processing of the section of the application.

25. (Previously Presented) The hardware assisted dynamic optimizer of claim 21, wherein the hardware assisted dynamic optimizer obtains the captured profiles of the one or more microarchitecture events from the second level profile buffer.

26. (Canceled)

27. (Previously Presented) The hardware assisted dynamic optimizer of claim 21, wherein the hardware assisted dynamic optimizer sets conditions for transferring captured profiles from the first level profile buffer to the second level profile buffer.

28. (Previously Presented) The hardware assisted dynamic optimizer of claim 27, wherein the hardware assisted dynamic optimizer sets one or more conditions for obtaining captured profiles from the second level profile buffer.

29. (Previously Presented) The hardware assisted dynamic optimizer of claim 28, wherein a memory buffer in the second level profile buffer is assigned to a microarchitecture event, and wherein the hardware assisted dynamic optimizer accesses the profiles of the microarchitecture event when the memory buffer assigned to the microarchitecture event is fully allocated or when a condition for obtaining captured profiles is met.

30. (Original) The hardware assisted dynamic optimizer of claim 29, wherein the hardware assisted dynamic optimizer accesses the profiles of a microarchitecture event upon receiving an interrupt or special event handler.

Please add the following new claims:

31. (New) The method of claim 1, wherein captured profile data is treated as an operand of an instruction for writing back to the first level profile buffer.

32. (New) The medium of claim 11, wherein captured profile data is treated as an operand of an instruction for writing back to the first level profile buffer.

33. (New) The hardware assisted dynamic optimizer of claim 21, wherein a captured profile is treated as an operand of a instruction for writing back to the first level profile buffer.